**Перегрузка операций**

Настоящий класс ostream определяет операцию <<, чтобы сделать удобным вывод нескольких объектов одним оператором. Давайте посмотрим, как это сделано.

Чтобы определить @, где @ - некоторая операция языка C++, для каждого определяемого пользователем типа вы определяете функцию с именем operator@, которая получает параметры соответствующего типа. Например:

class ostream {

//...

ostream operator<<(char\*);

};

ostream ostream::operator<<(char\* p)

{

while (\*p) buf.sputc(\*p++);

return \*this;

}

определяет операцию << как член класса ostream, поэтому s< ");

а если применить операцию взятия адреса, то вы получите адрес объекта, на который ссылается ссылка:

&s1 == &my\_out

Первая очевидная польза от ссылок состоит в том, чтобы обеспечить передачу адреса объекта, а не самого объекта, в функцию вывода (в некоторых языках это называется передачей параметра по ссылке):

ostream& operator<<(ostream& s, complex z) {

return s << "(" << z.real << "," << z.imag << ")";

}

Достаточно интересно, что тело функции осталось без изменений, но если вы будете осуществлять присваивание s, то будете воздействовать на сам объект, а не на его копию. В данном случае то, что возвращается ссылка, также повышает эффективность, поскольку очевидный способ реализации ссылки - это указатель, а передача указателя гораздо дешевле, чем передача большой структуры данных.

Ссылки также существенны для определения потока ввода, поскольку операция ввода получает в качестве операнда переменную для считывания. Если бы ссылки не использовались, то пользователь должен был бы явно передавать указатели в функции ввода.

class istream {

//...

int state;

public:

istream& operator>>(char&ap;);

istream& operator>>(char\*);

istream& operator>>(int&);

istream& operator>>(long&);

//...

};

Заметьте, что для чтения long и int используются разные функции, тогда как для их печати требовалась только одна. Это вполне обычно, и причина в том, что int может быть преобразовано в long по стандартным правилам неявного преобразования, избавляя таким образом программиста от беспокойства по поводу написания обеих функций ввода.